CMP205: Computer Graphics

CAIRO UNIVERSITY

Lecture 7: Surface Shading

Mohamed Alaa El-Dien Aly
Computer Engineering Department
Cairo University
Fall 2012

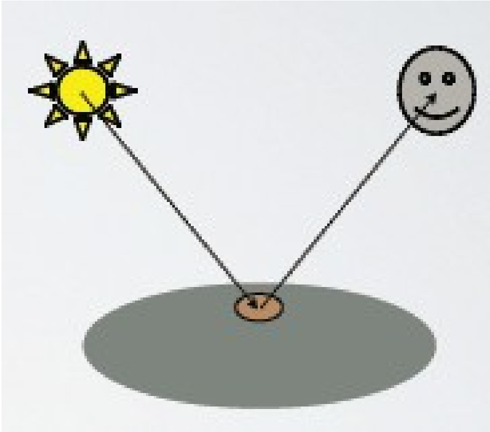
Agenda

- Lighting and Surface Rendering
- Shading Models
  - Diffuse
  - Ambient
  - Specular
- Light Sources
- Surface Rendering
  - Flat
  - Gourard
  - Phong

Acknowledgment: Some slides adapted from Steve Marschner and Maneesh Agrawala

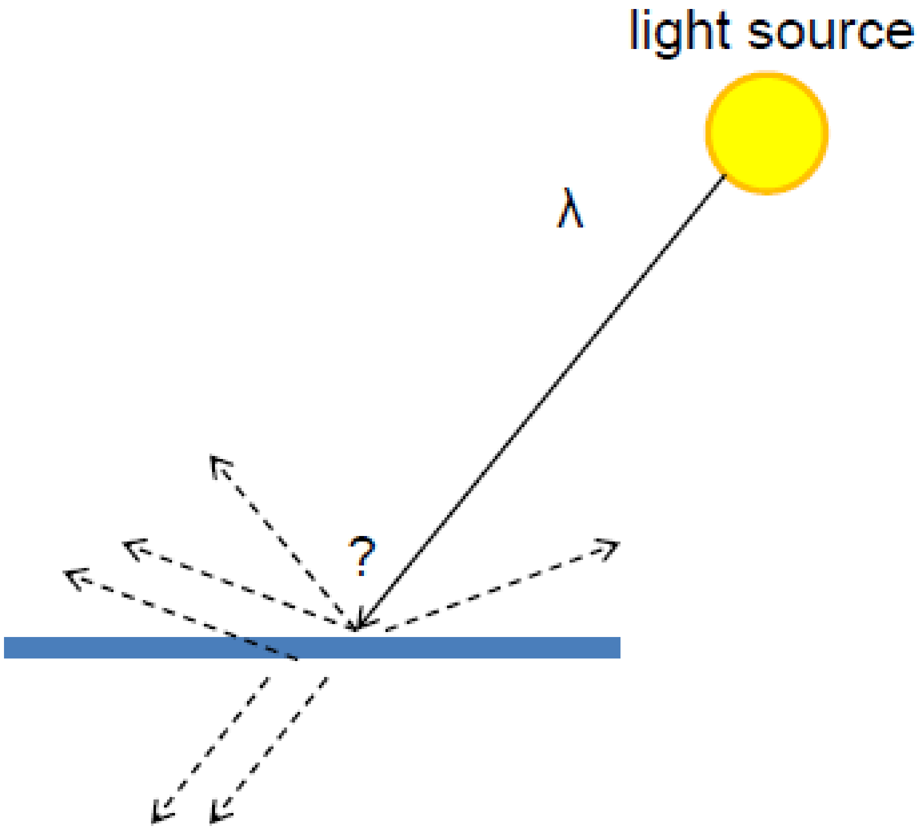
Lighting

- **Lighting Model**: what is the color of a particular position on the object surface
  - a.k.a.: Shading Model, Illumination Model

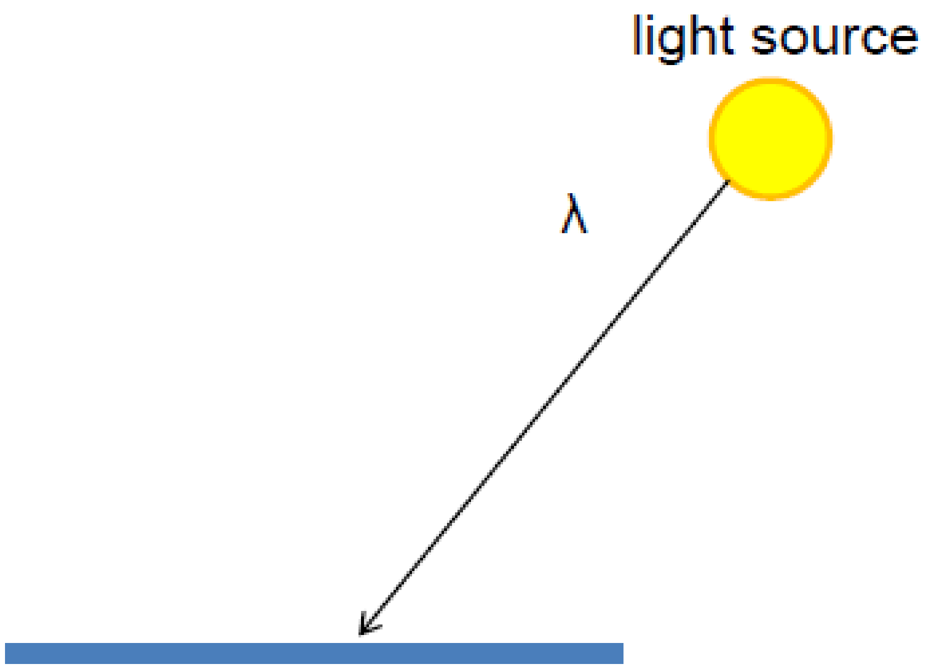


- **Surface Rendering Model**: what is the color of a pixel of a rasterized triangle
  - a.k.a.: Shading

Light and Surfaces

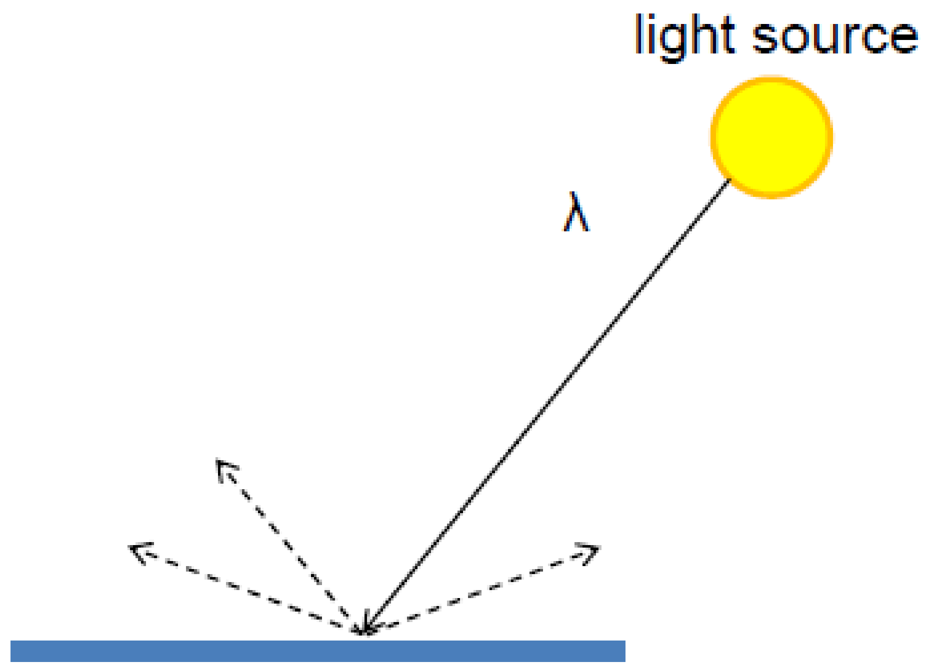


Light and Surfaces



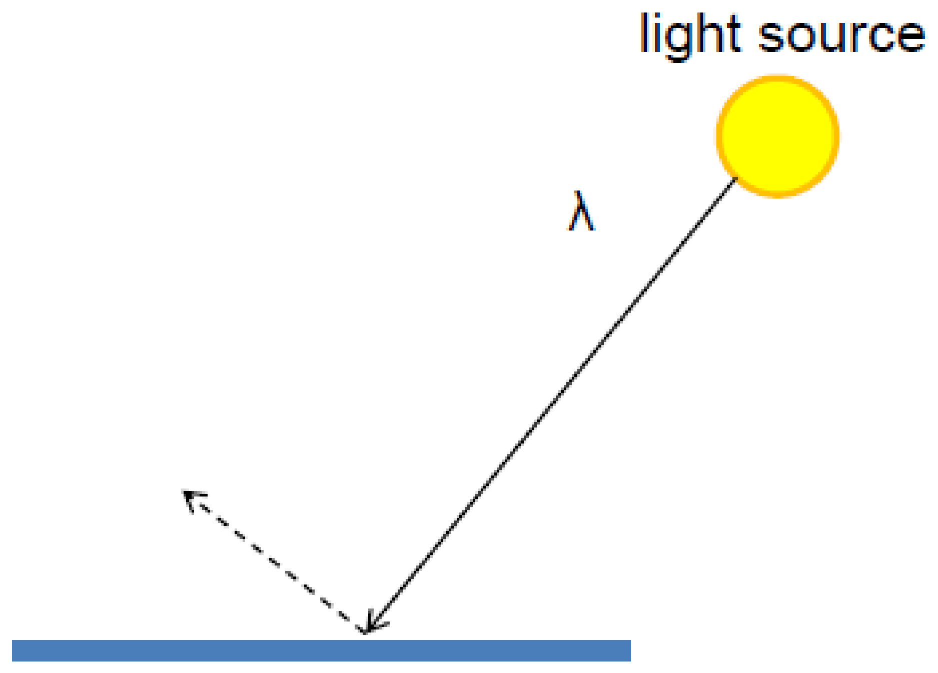
Absorption

Light and Surfaces



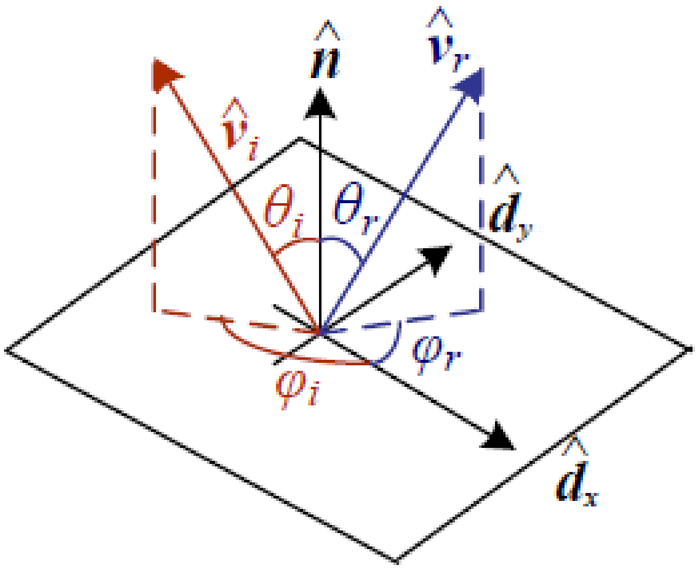
Diffusion

Light and Surfaces



(Specular) Reflection

Bidirectional Reflectance Distribution Function (BRDF)



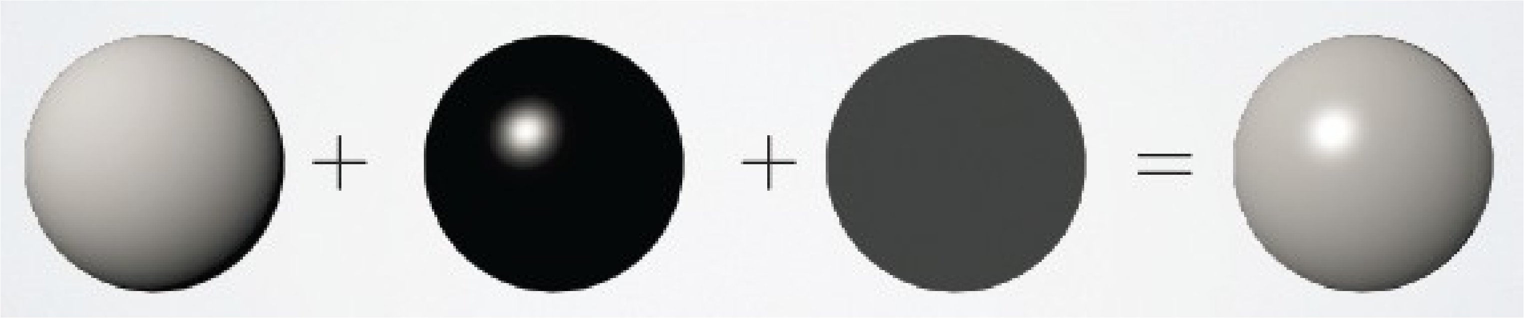
$$\rho\left(\hat{v}_i, \hat{v}_r, \hat{n}\right)$$

Ratio between reflected and incident light

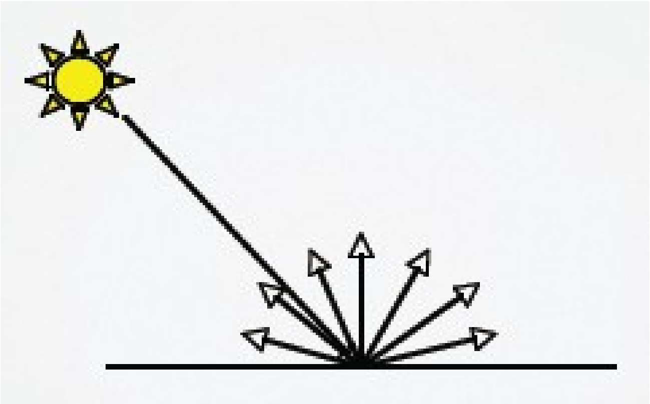
BRDF

Approximate BRDF as:

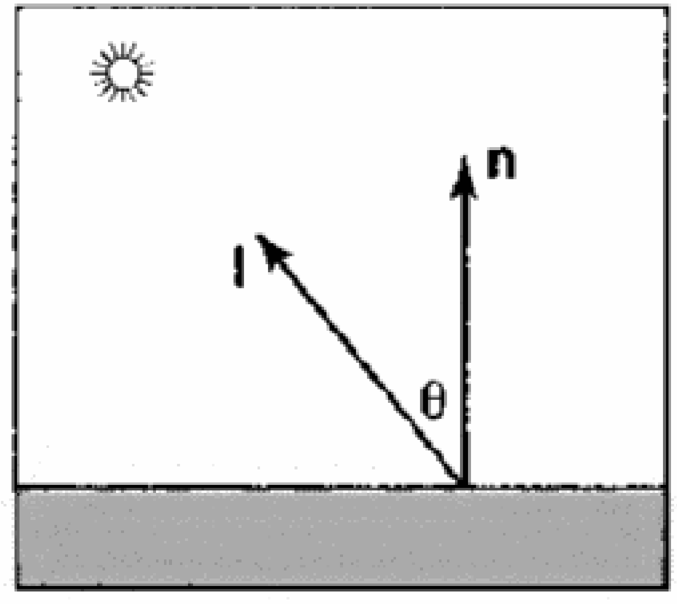
- A diffuse component
- A specular component
- An ambient component



Diffuse Shading



Reflected light same in all directions

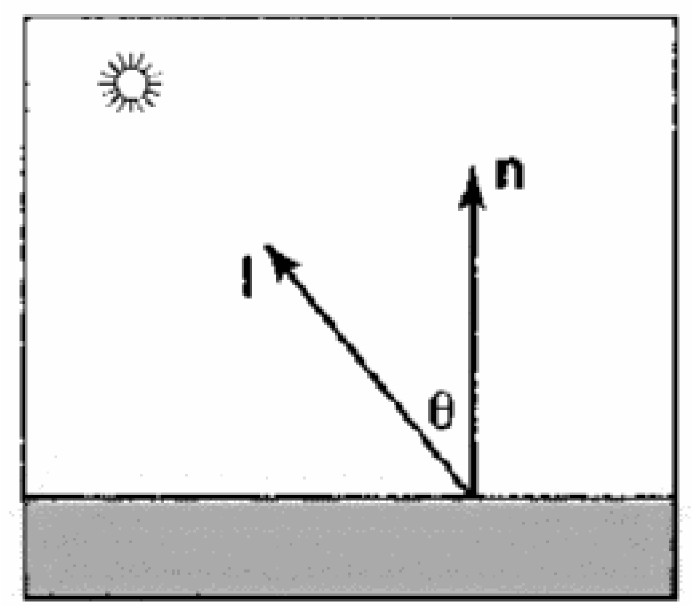


Reflected light depends on θ

Diffuse Shading

Lambert's Cosine Law

$$\rho_d \propto \cos\theta \quad \text{or} \quad \rho_d \propto \boldsymbol{n}\cdot\boldsymbol{l}$$



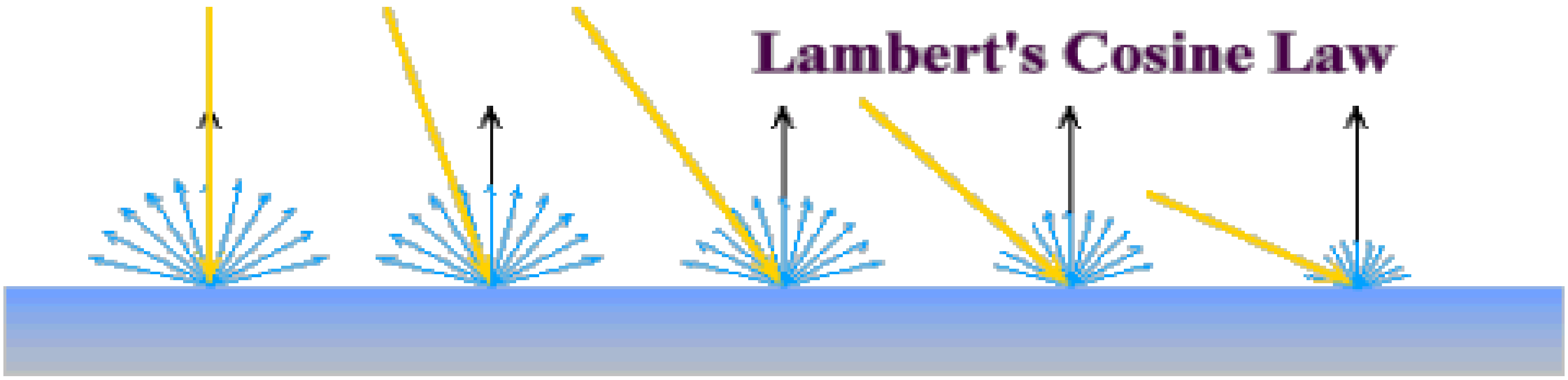
$$\rho_d = k_d (\boldsymbol{n}\cdot\boldsymbol{l})$$

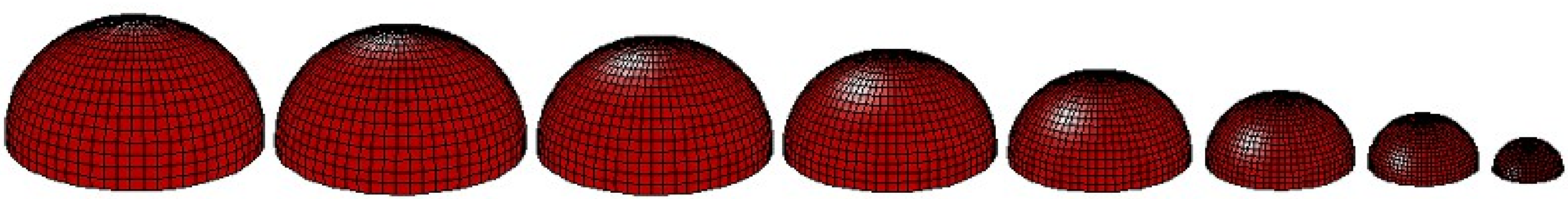
What if *cos* is negative?

$$\rho_d = k_d \max(0, \boldsymbol{n}\cdot\boldsymbol{l})$$

$$R = k_d I \max(0, \boldsymbol{n}\cdot\boldsymbol{l})$$

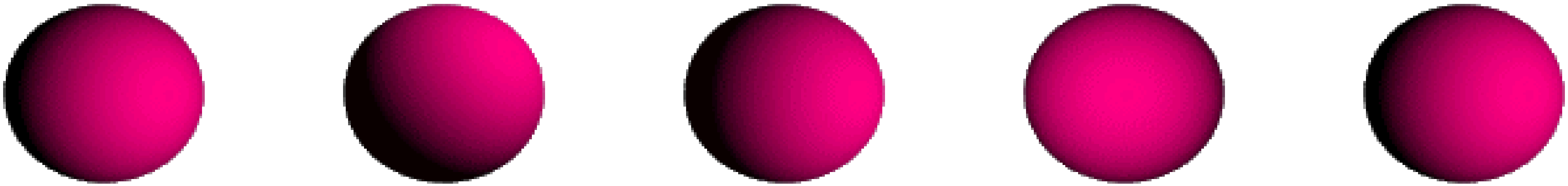
Diffuse Shading





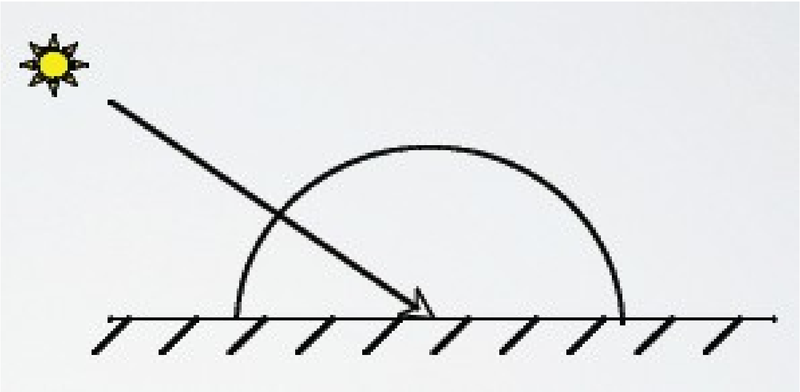
Reflected light independent of viewing direction !

Diffuse Shading

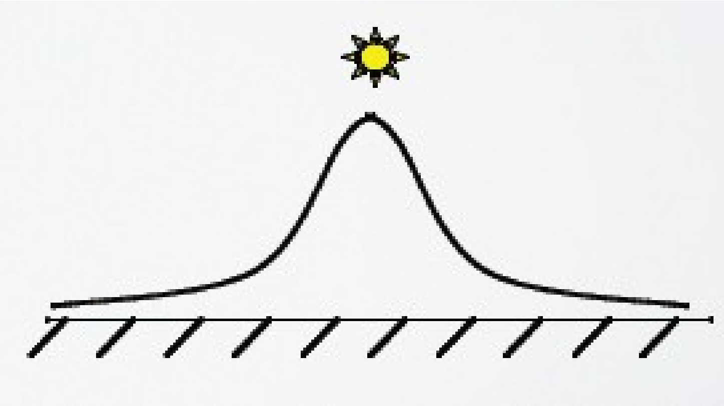


Reflected light depends on position of light source !

Diffuse Shading

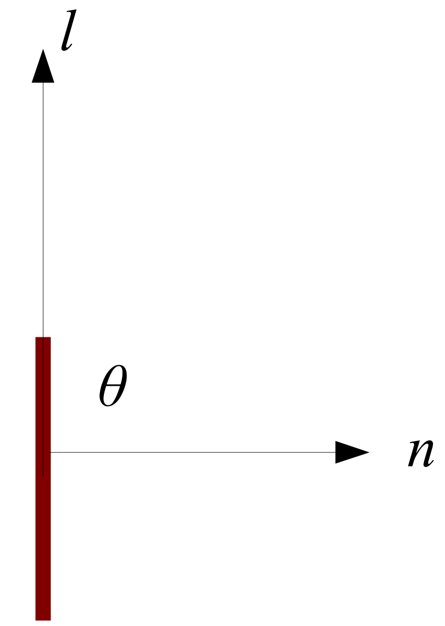


Light leaving a surface point in a specific direction



Light leaving each point on the surface

Ambient Shading



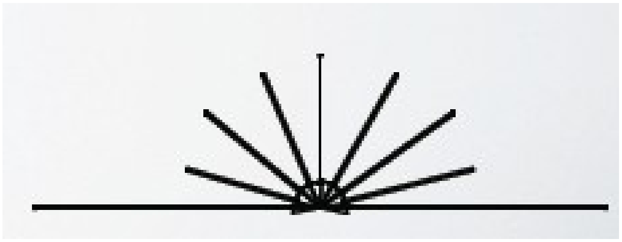
$$\rho_d = k_d \max(0, \boldsymbol{n} \cdot \boldsymbol{l})$$

What if $\theta \geq 90$?

$$\rho_d = 0 \text{ i.e. dark surface}$$

Add *ambient* lighting component.
Accounts for light reflected from the surroundings.

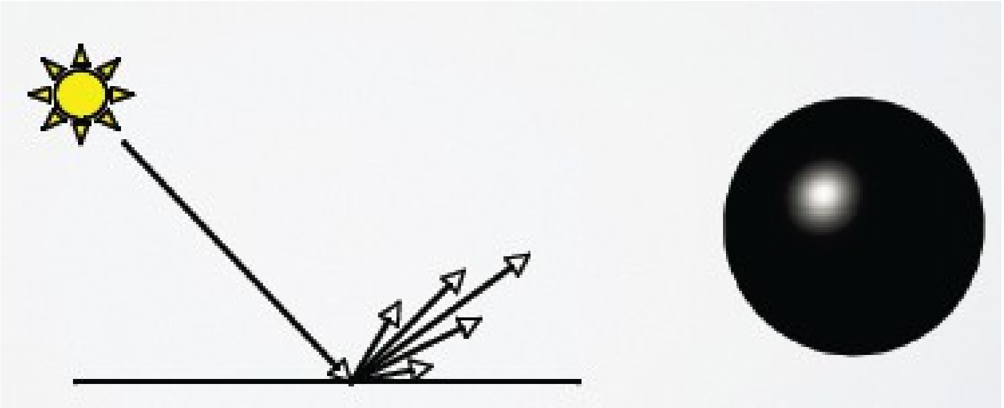
$$\rho_a = k_a$$



$$R = k_a I_a$$

Specular Shading

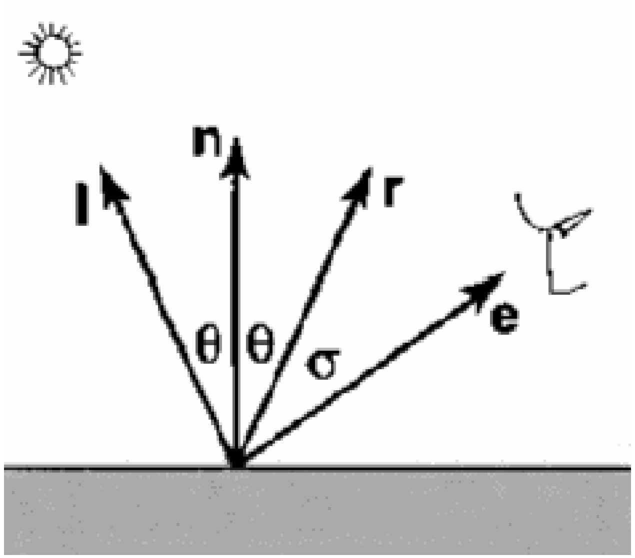
- Mirror-like reflection
- Good approximation for some surfaces
- Depends on the viewing direction
- Phong Ilumination Model



Specular Shading

Incidence angle equals Reflection angle

Specular highlight depends on viewing angle σ



$$\rho_s = k_s \max\left(0, \boldsymbol{e}\cdot\boldsymbol{r}\right)$$

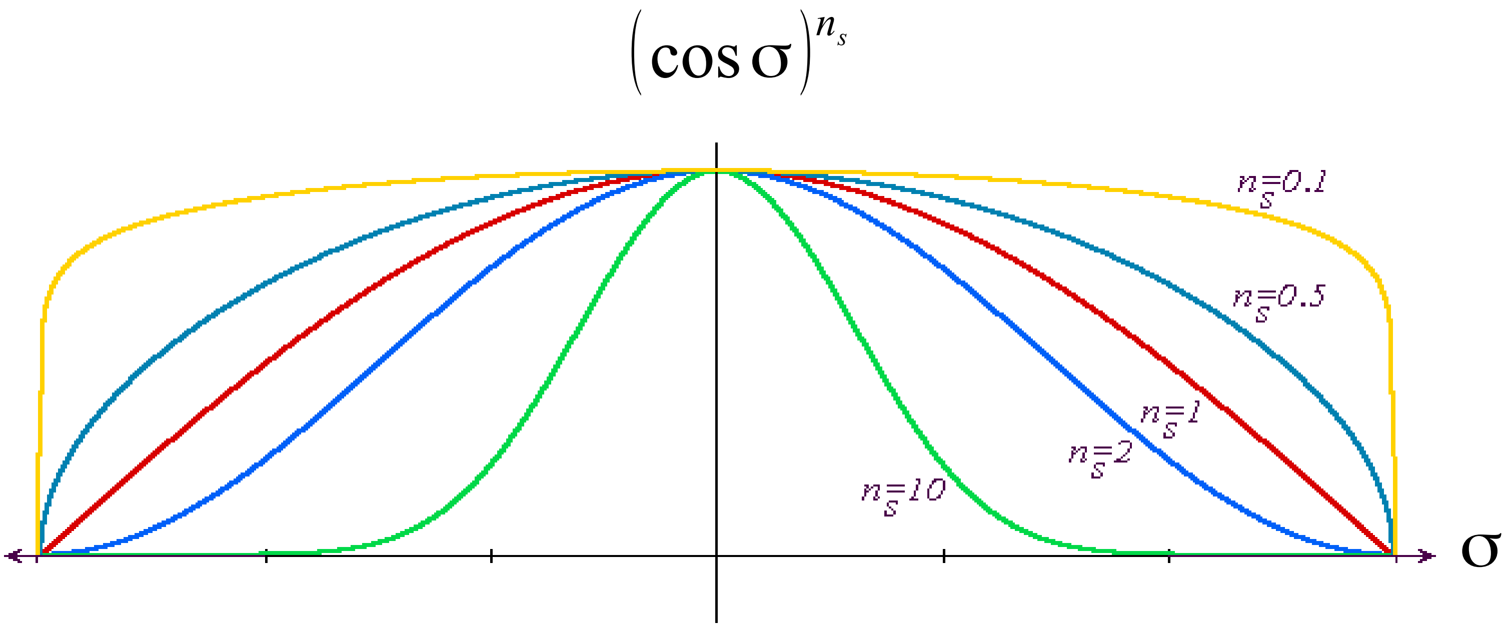
Problems?

$$\rho_s = k_s \max\left(0, \boldsymbol{e}\cdot\boldsymbol{r}\right)^p$$

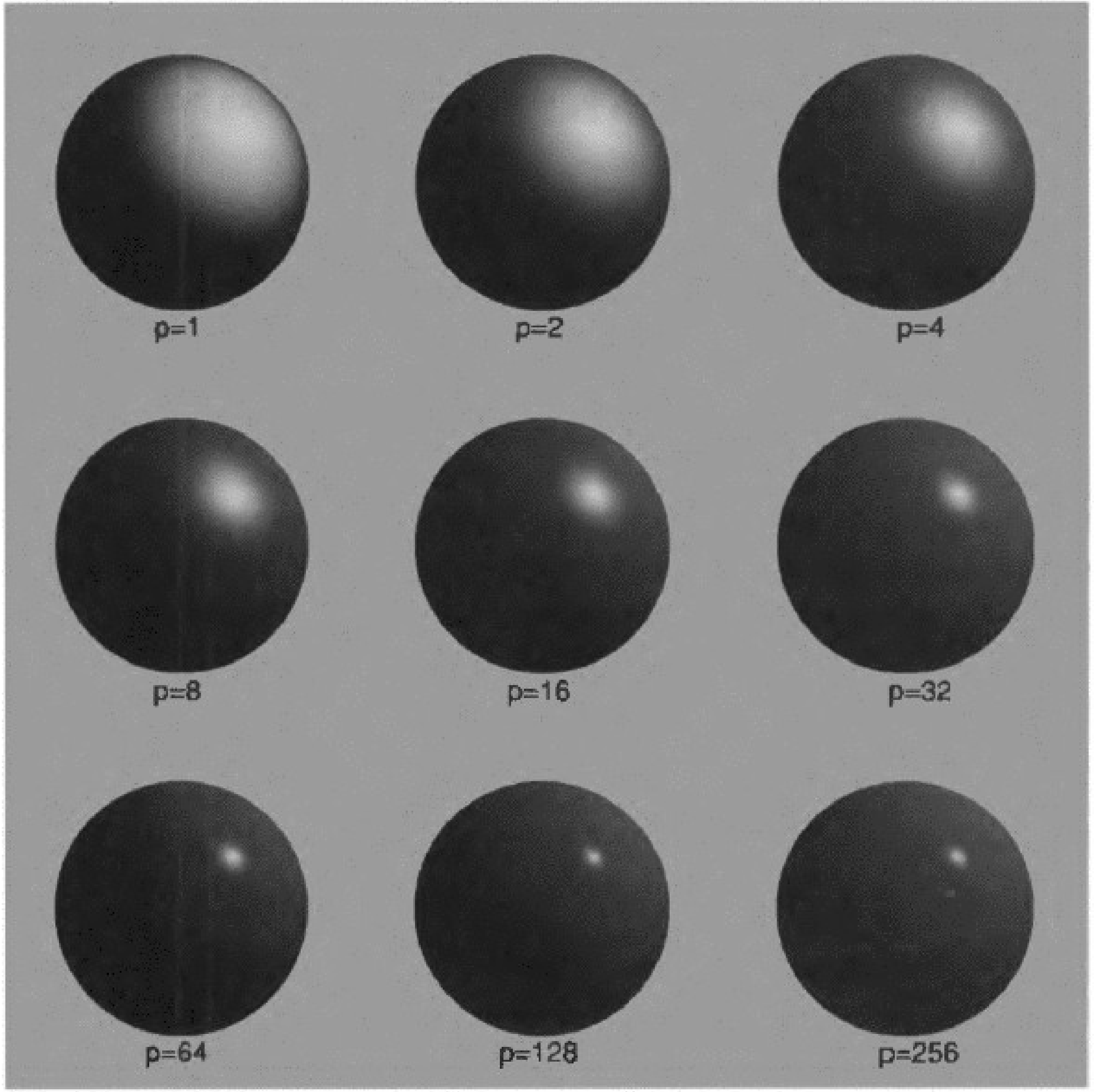
p: Phong Exponent

$$R = k_s I \max\left(0, \boldsymbol{e}\cdot\boldsymbol{r}\right)^p$$

Specular Shading

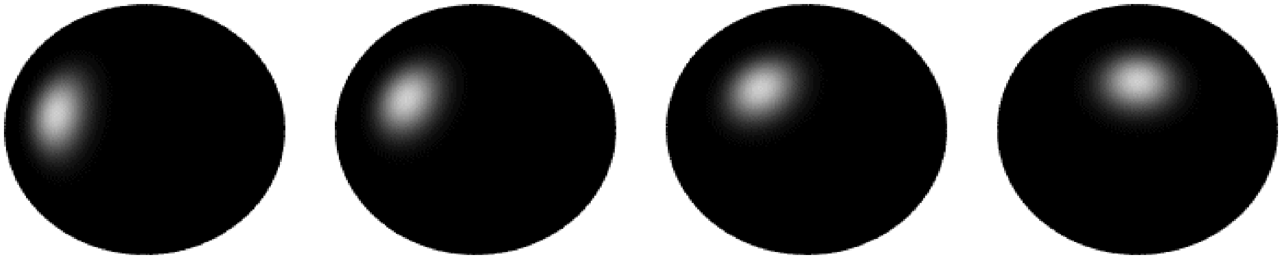


Specular Shading

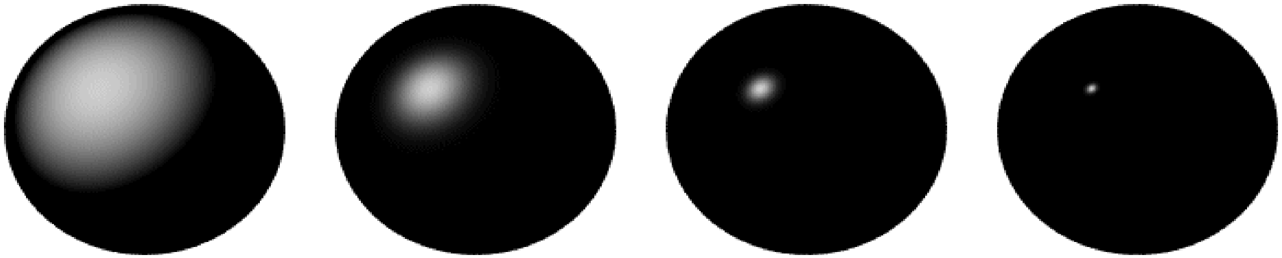


Different values for p

Specular Shading



Different light source direction

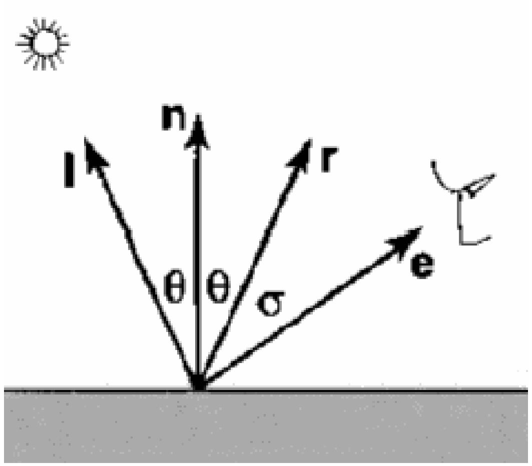


Different values for p

Specular Shading

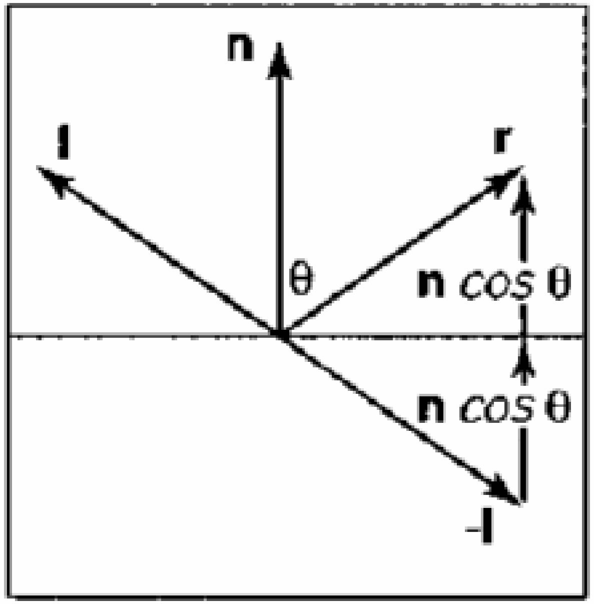
$$\rho_s = k_s \max\left(0, \boldsymbol{e} \cdot \boldsymbol{r}\right)^p$$

How do we compute r?



$$r = -\boldsymbol{l} + 2\cos\theta\,\boldsymbol{n}$$

$$r = -\boldsymbol{l} + 2\left(\boldsymbol{l} \cdot \boldsymbol{n}\right)\boldsymbol{n}$$

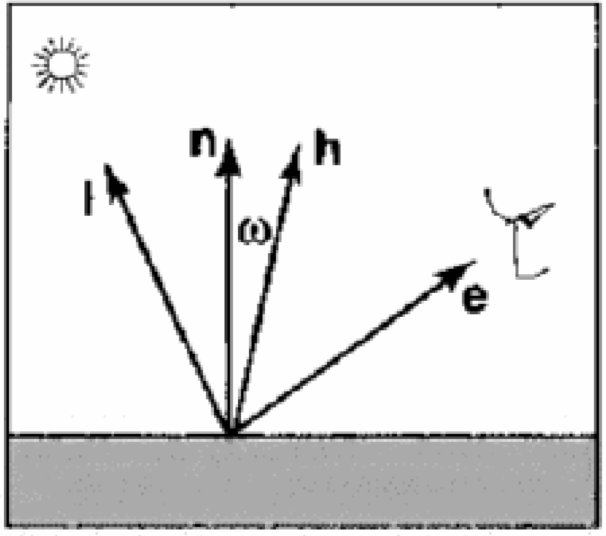


Specular Shading

Alternative: Look at halfway vector $\boldsymbol{h}$

Want $\boldsymbol{h}$ to line up with $\boldsymbol{n}$ i.e. $\omega = 0$

$$\rho_s = k_s (\boldsymbol{h} \cdot \boldsymbol{n})^p \quad \text{or} \quad R = k_s I (\boldsymbol{h} \cdot \boldsymbol{n})^p$$



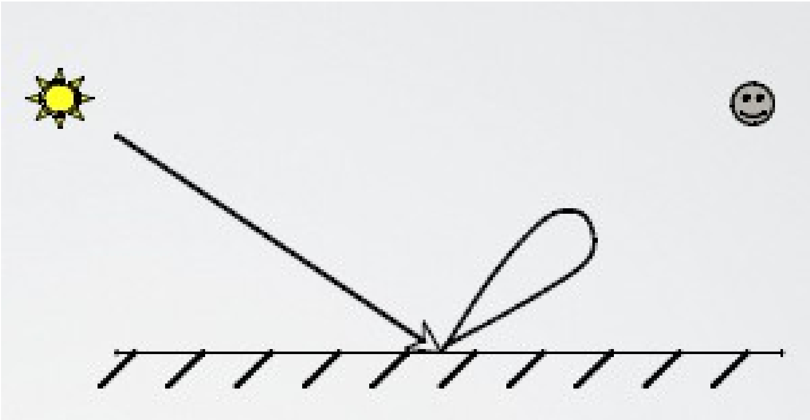
What is $\boldsymbol{h}$?

$$\boldsymbol{h} = \frac{\boldsymbol{l} + \boldsymbol{e}}{\|\boldsymbol{l} + \boldsymbol{e}\|}$$

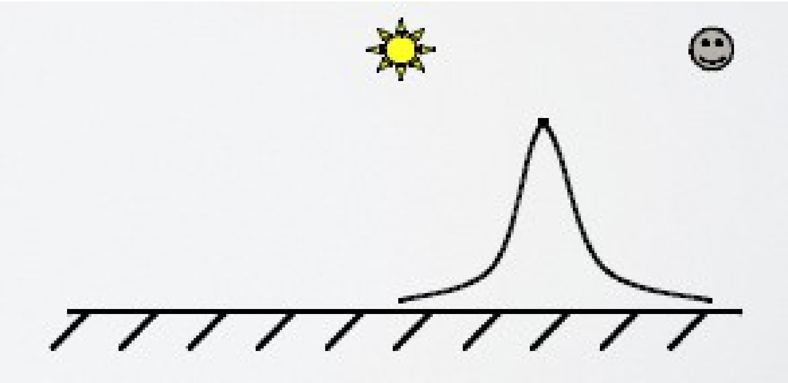
Advantage?
Dot product always +ve above the plane!

Disadvantage?
Square root and divide !

Specular Shading

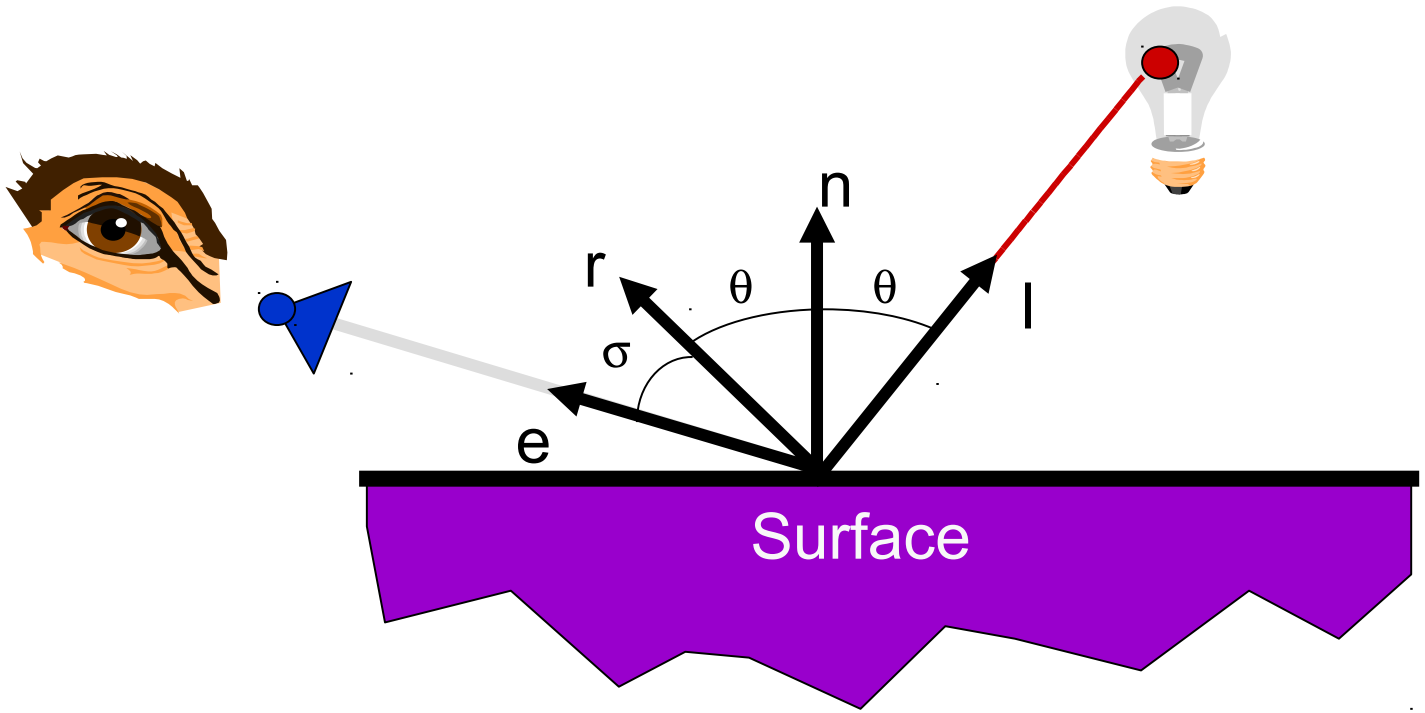


Light leaving a surface point in a specific direction



Light leaving each point on the surface

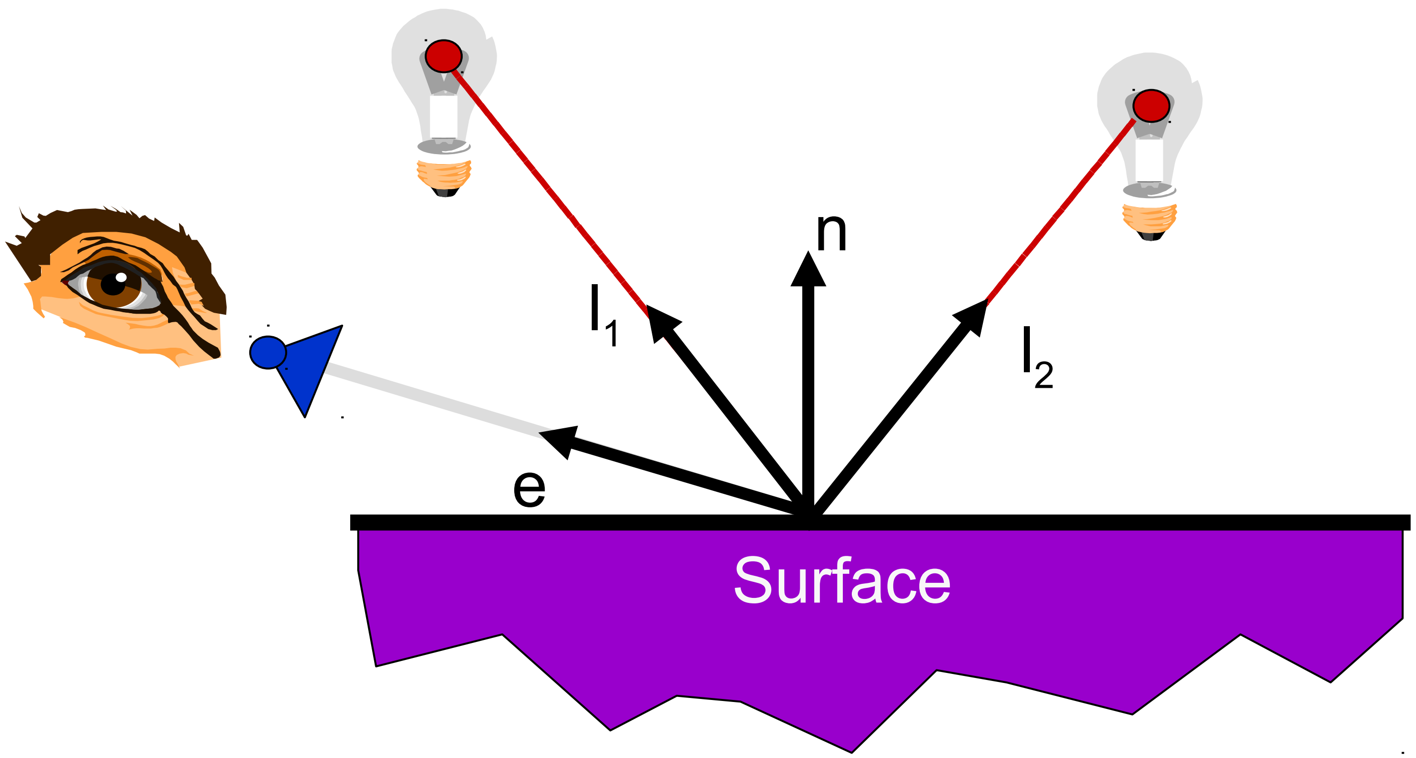
Summing Up: Phone Shading Model



$$R = k_a I_a + k_d I \max(0, \boldsymbol{l} \cdot \boldsymbol{n}) + k_s I \max(0, \boldsymbol{e} \cdot \boldsymbol{r})^p$$

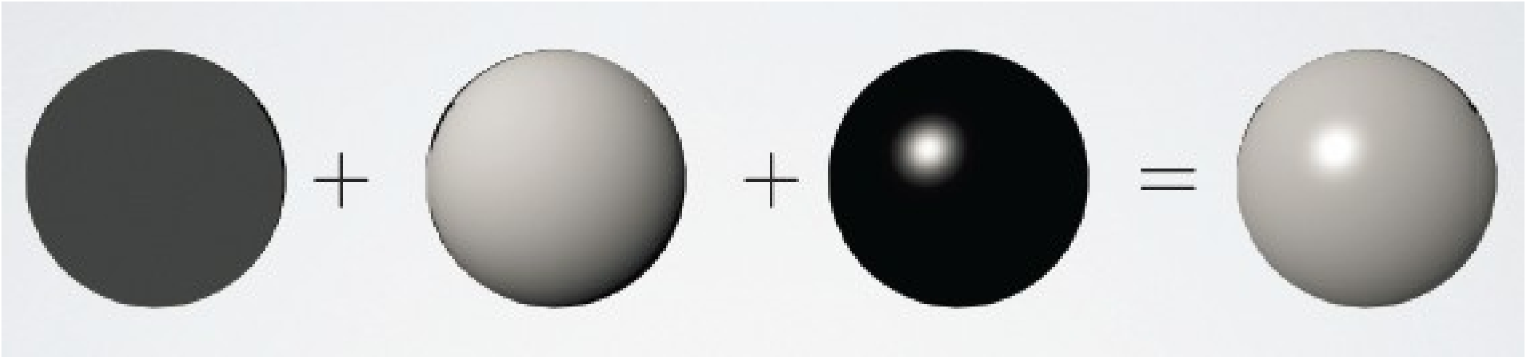
R: Reflected light
I: Incident light source

Summing Up: Phone Shading Model

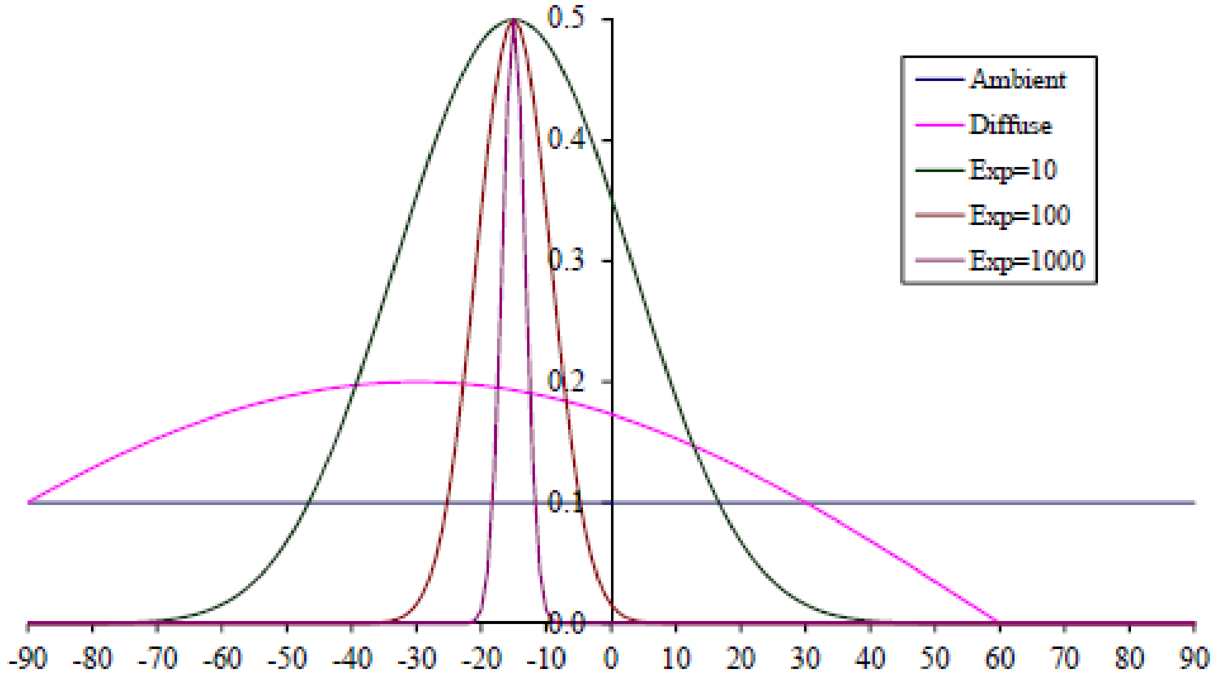


$$R = k_a I_a + \sum_i \left[k_d I_i \max\left(0, \boldsymbol{l_i} \cdot \boldsymbol{n}\right) + k_s I_i \max\left(0, \boldsymbol{e} \cdot \boldsymbol{r_i}\right)^p \right]$$

Summing Up: Phone Shading Model



Ambient Diffuse Specular



Color

What about colored light?

Create different components for R, G, and B !

For example for the blue component:

$$R_B = k_{aB} I_{aB} + \sum_i k_{dB} I_{iB} \max\left(0, \boldsymbol{l_i} \cdot \boldsymbol{n}\right) + k_{sB} I_{iB} \max\left(0, \boldsymbol{e} \cdot \boldsymbol{r_i}\right)^p$$

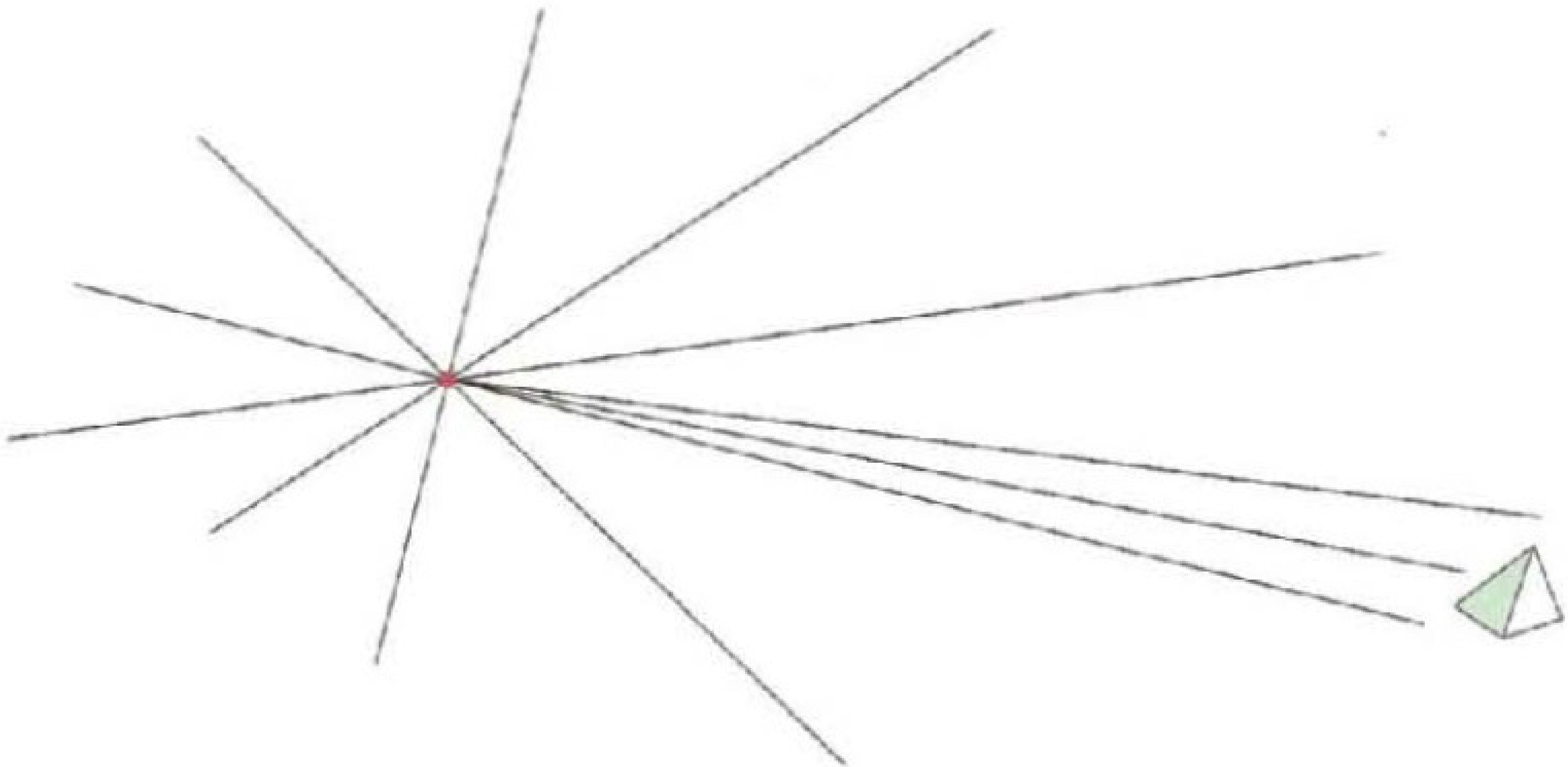
So we end up with 3 dimensional vectors for: $\boldsymbol{k_a}, \boldsymbol{k_d}, \boldsymbol{k_s}$

$$\boldsymbol{k_a} = \begin{bmatrix} k_{aR} \\ k_{aG} \\ k_{aB} \end{bmatrix} \;\&\; \boldsymbol{k_d} = \begin{bmatrix} k_{dR} \\ k_{dG} \\ k_{dB} \end{bmatrix} \;\&\; \boldsymbol{k_s} = \begin{bmatrix} k_{sR} \\ k_{sG} \\ k_{sB} \end{bmatrix} \quad \in R^3$$

Light Sources

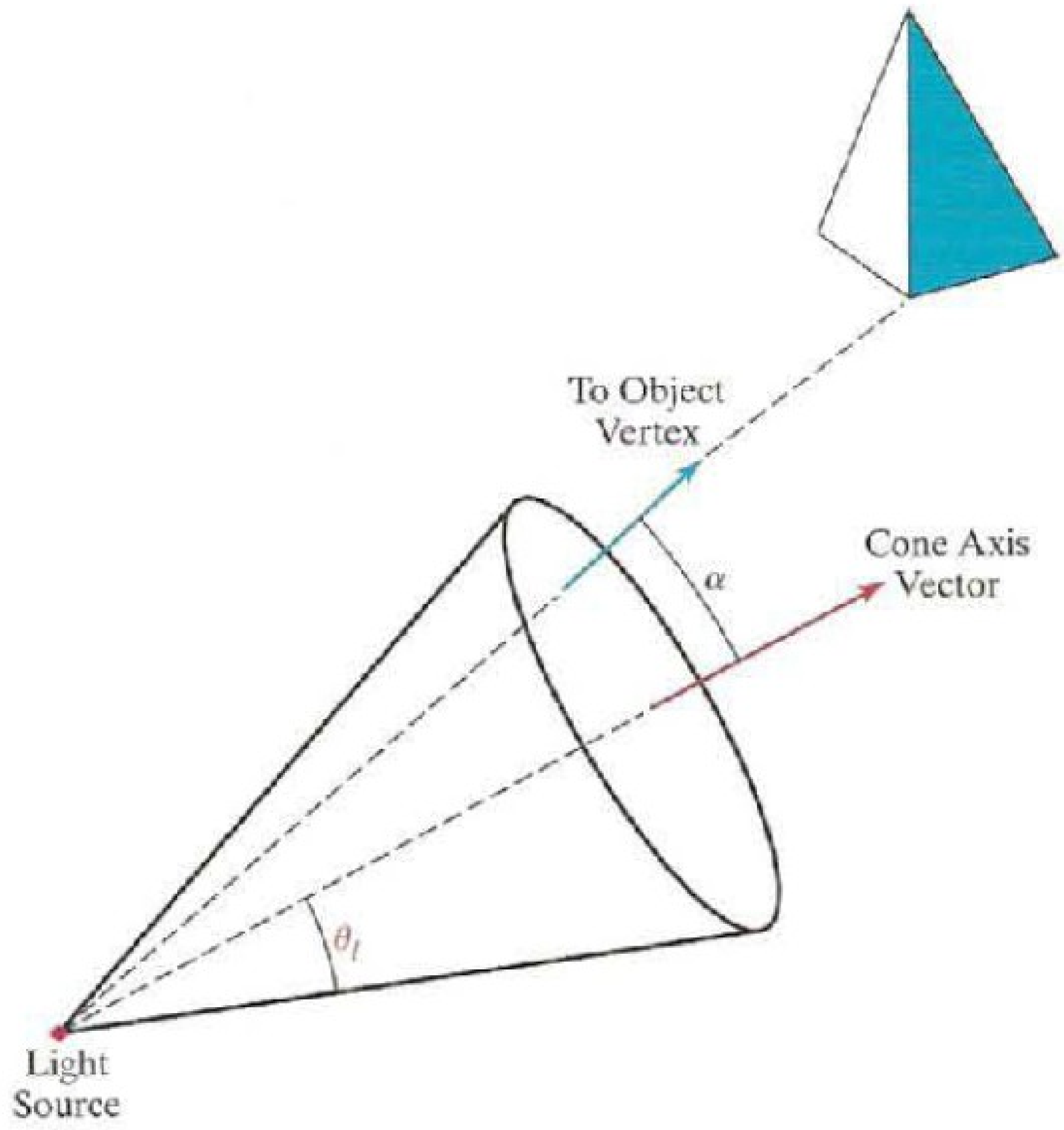
Point Light Source

Light Sources



Point Light Source at Infinity
Directional Light Source

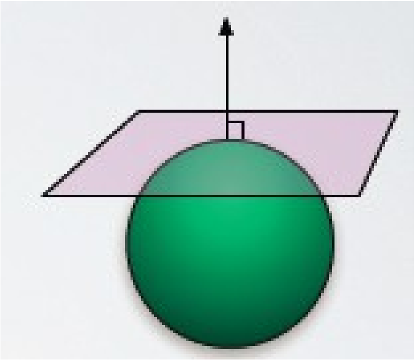
Light Sources



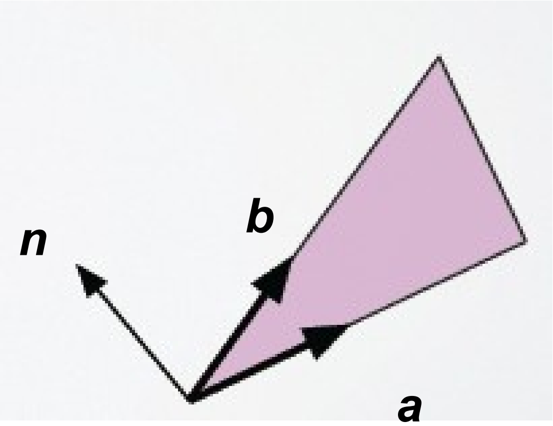
Spotlight Light Source

Surface Normals

Vector normal to all tangent vectors

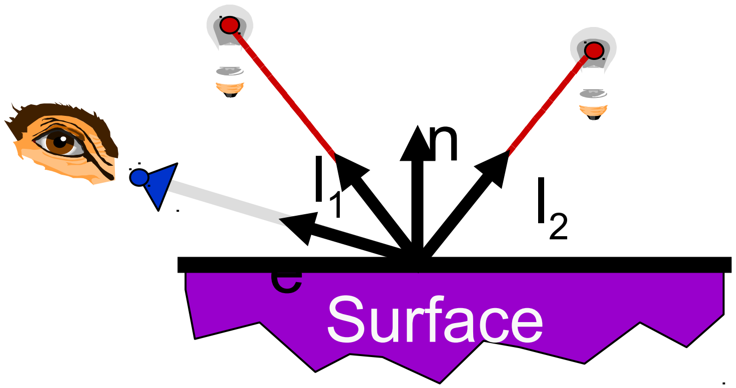


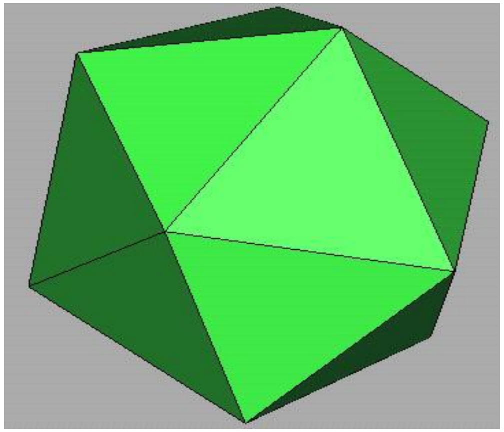
$$n = a \times b$$



Surface Rendering

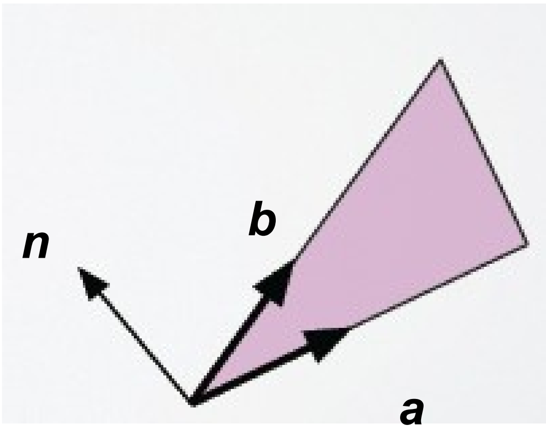
Now we can compute light reflected from any surface point





How can we rasterize a triangle to get pixel values ?

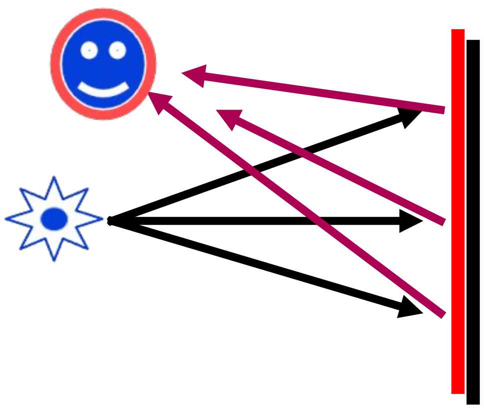
Flat Shading



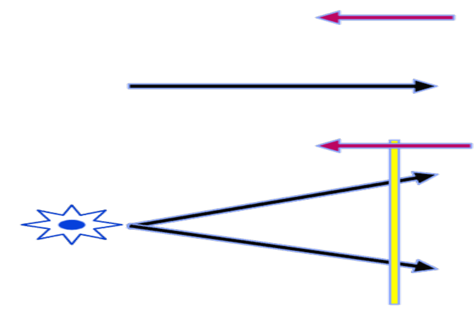
- Every triangle has only surface normal
- One computation per triangle
- One color per triangle

- Very cheap
- Faceted appearance
- Surfaces not smooth

Flat Shading



Viewing direction not constant !



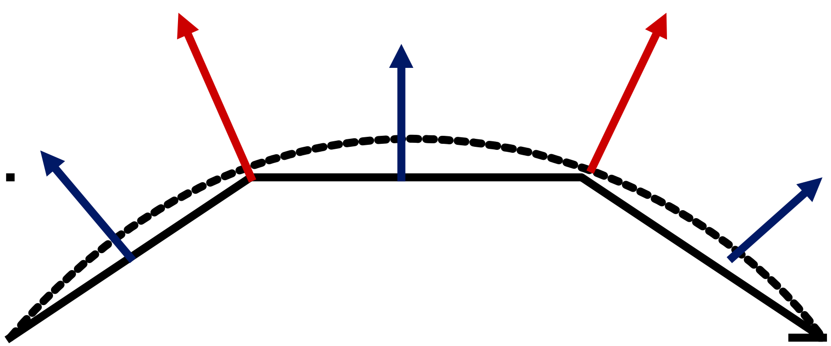
Light direction not constant !

Accurate when:

- Surface is already faceted
- Light source too far from surface
- Viewing direction too far from surface

Gourard Shading

- Normal vector at each vertex
- Can be
  - Average of face normals
  - Model supplied
- Used for shading

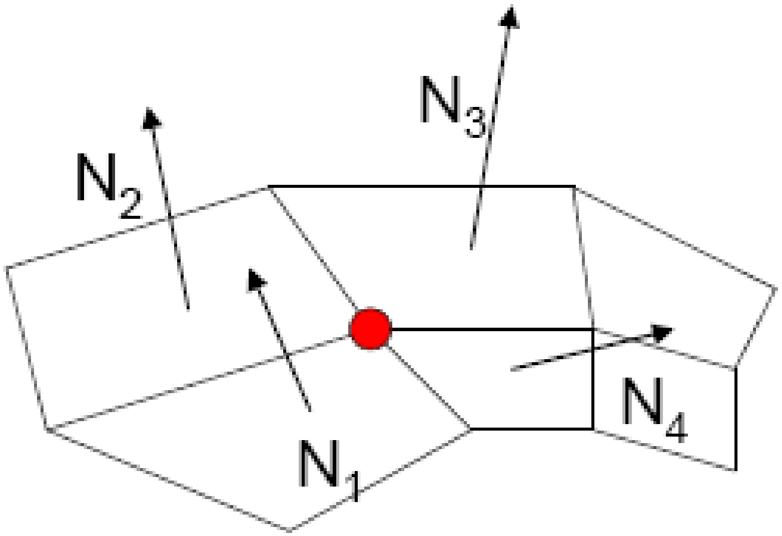


Vertex Normals

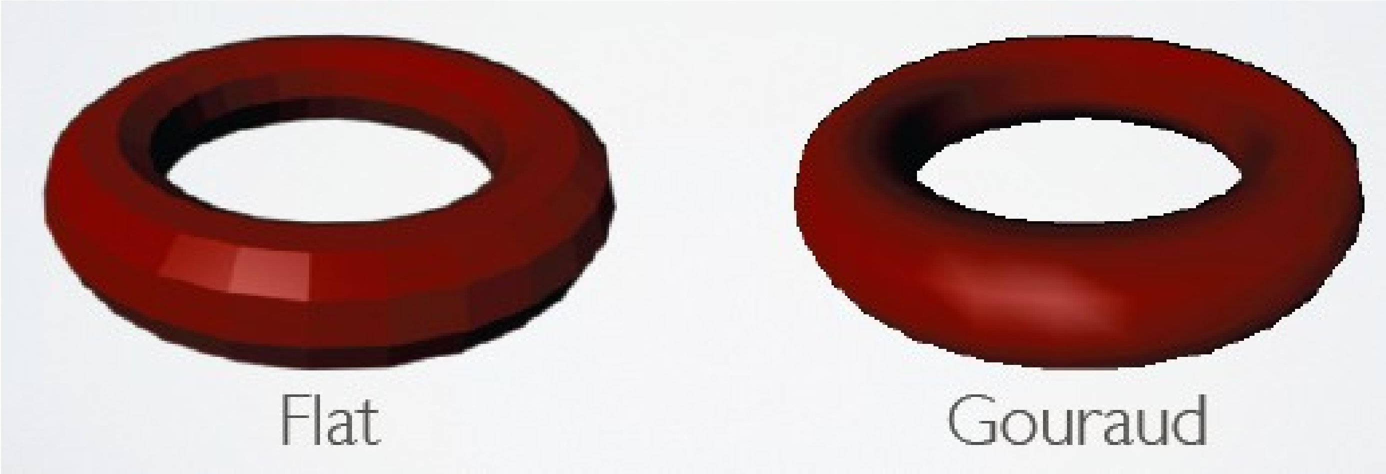
Gourard Shading

- Compute shading at each vertex using vertex normal
- Interpolate across triangle using Barycentric Coordinates

- Pros
  - Better than flat
  - Fast
- Cons
  - Bad speculars
  - Mach bands

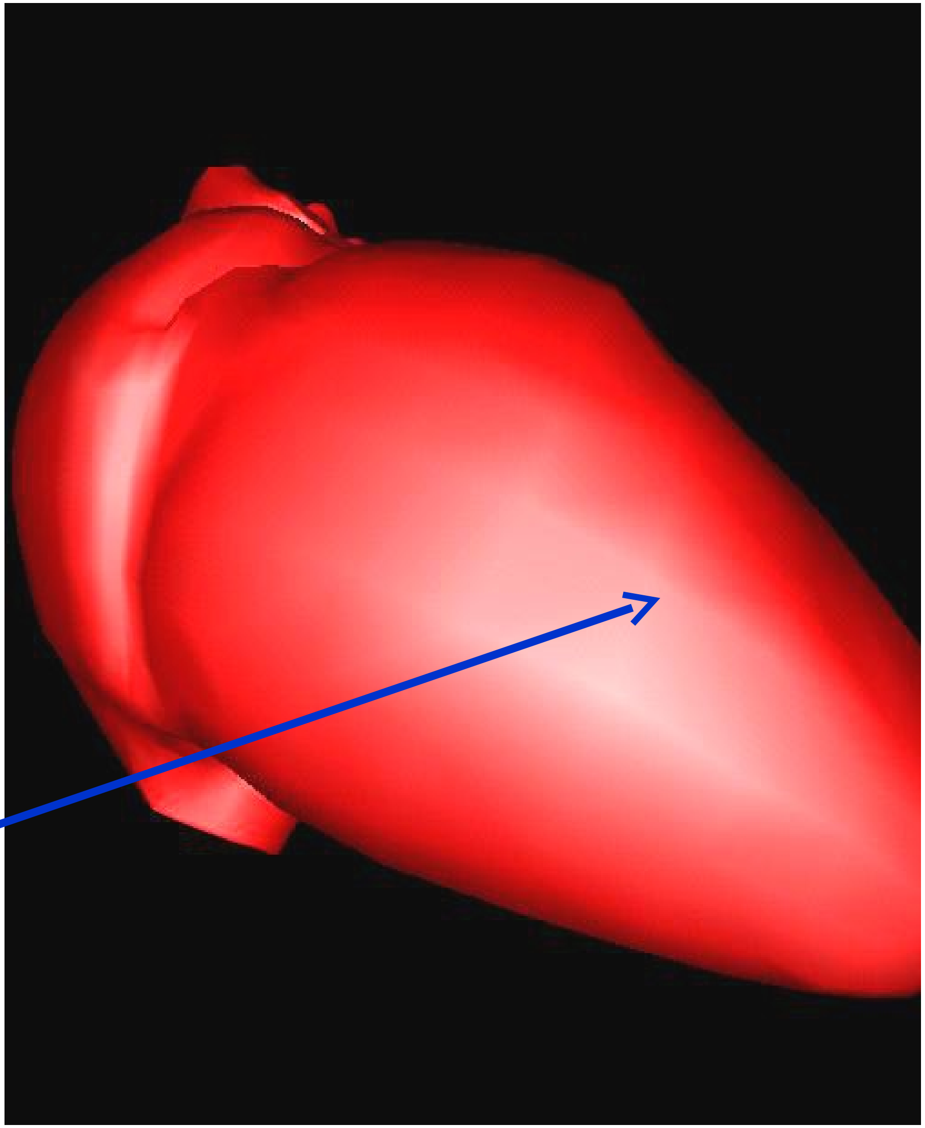


Gourard Shading



Gourard shading

Mach Banding

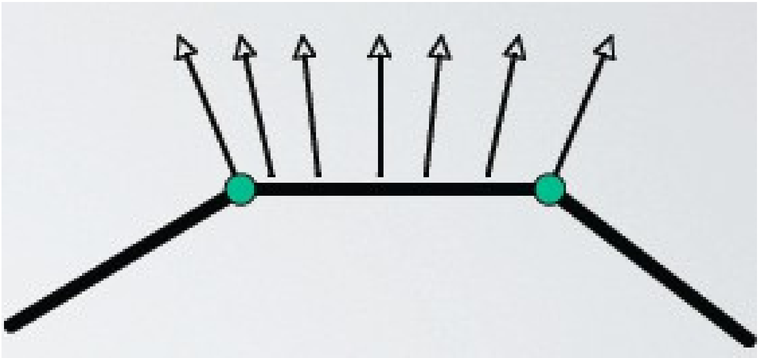


Discontinuities

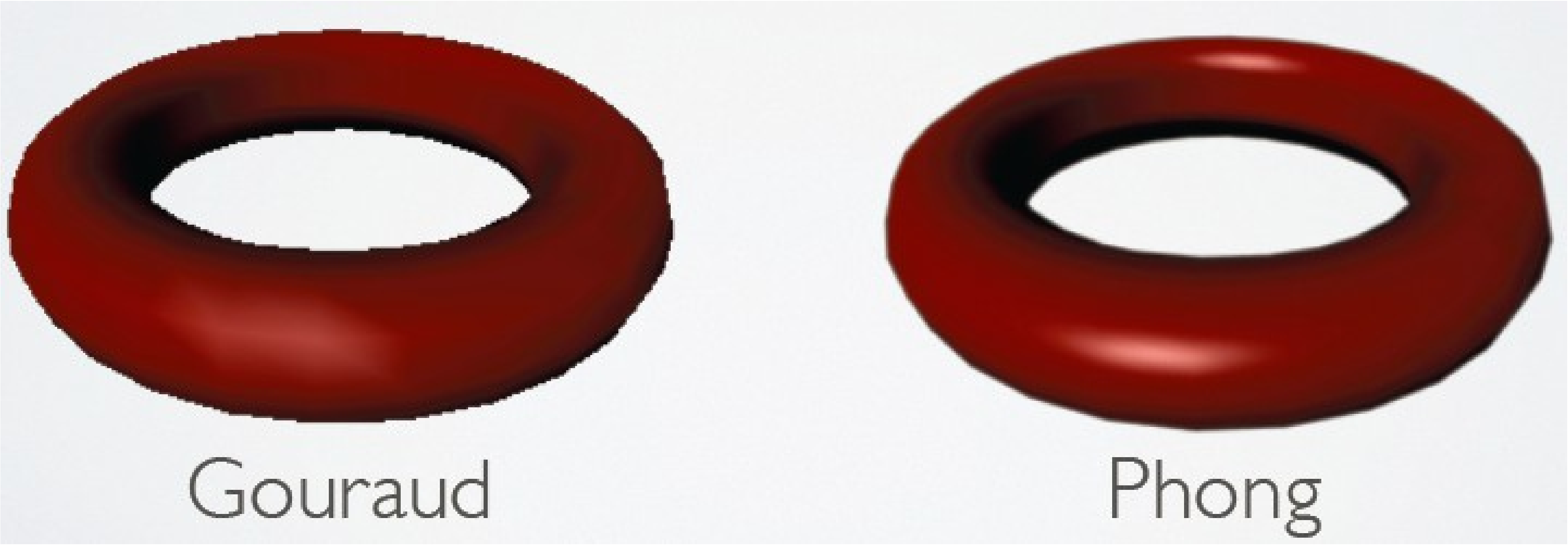
http://www.edcenter.sdsu.edu/slides/GA/visteacher/

Phong Shading

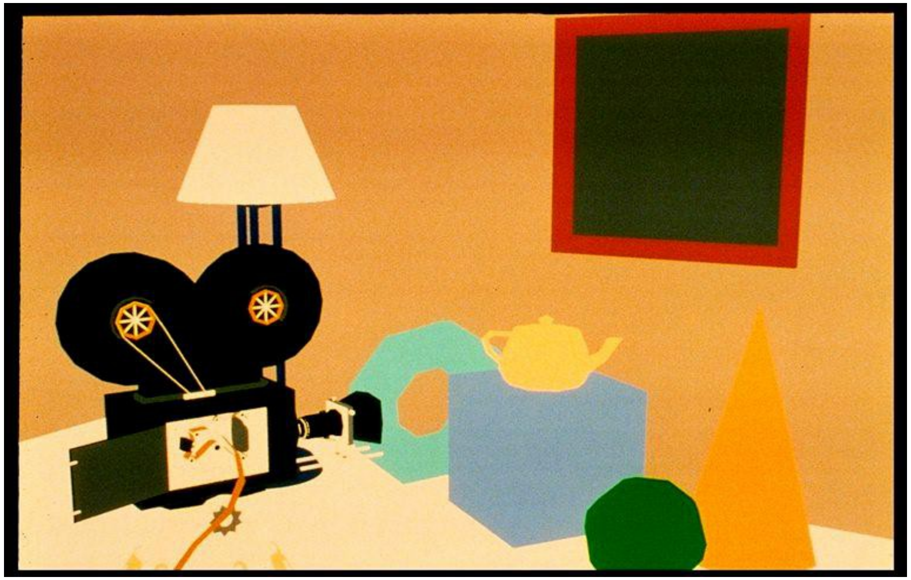
- Interpolate surface normals at each pixel *not* intensities. *How?*
- Compute shading at each pixel
- Very expensive!



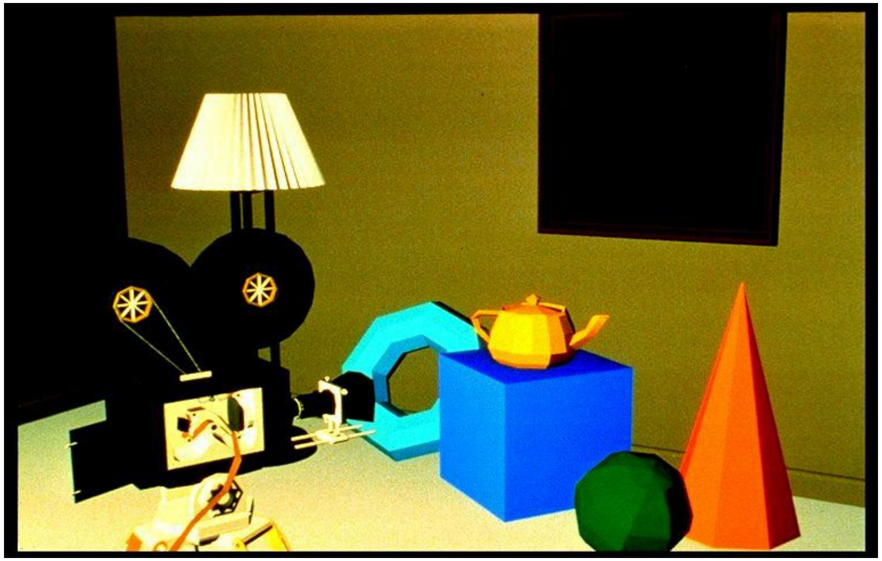
Phong Shading



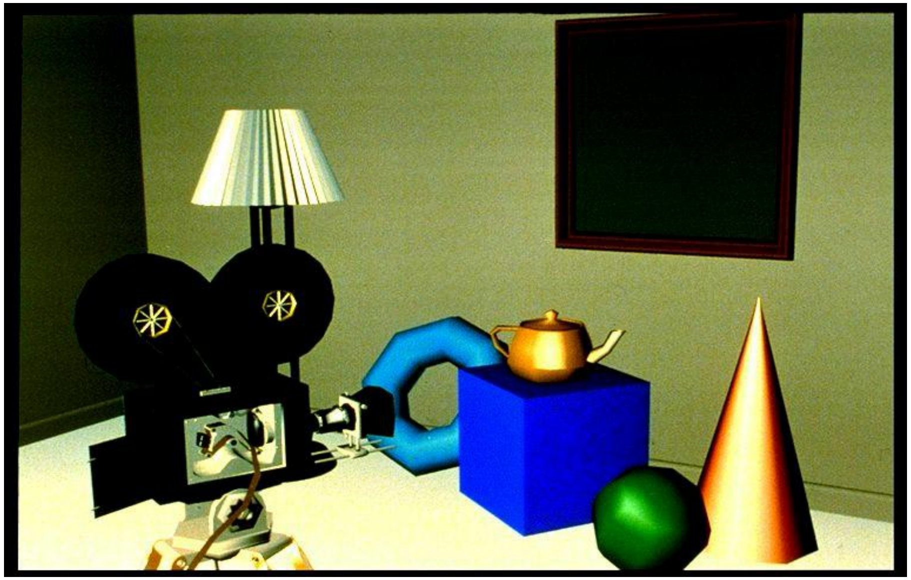
Surface Shading



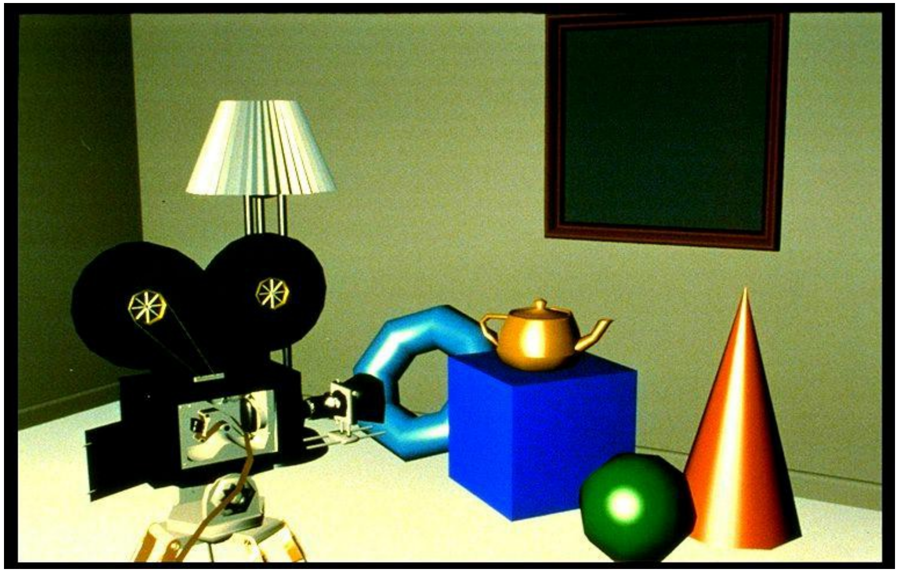
Ambient



Flat Shading



Gourard Shading



Phong Shading

Recap

- Lighting and Surface Rendering
- Shading Models
  - Diffuse
  - Ambient
  - Specular
- Light Sources
- Surface Rendering
  - Flat
  - Gourard
  - Phong